Fine-Tuned MobileNet With TensorFlow's Keras API for a data set of sign language digits

- By: Ismail Ouahbi (06/07/2023)

Introduction

`MobileNetV2` is a popular deep learning model that was introduced `by Google in 2018` as an improvement over its predecessor, `MobileNetV1`. It is `designed specifically for mobile and embedded devices with limited computational resources`.

The primary goal of MobileNetV2 is to provide a `lightweight and efficient architecture` while still achieving competitive accuracy in various computer vision tasks.

Why MobileNetV2?

Comparison of MobileNet versions

In both of our models, we use different versions of MobileNet models. MobileNet V2 is mostly a updated version of V1 that makes it even more efficient and powerful in terms of performance. We will see a few factor between both the models:

Version	MACs(millions)	Parameters(millions)
MobileNet V1	569	4.24
MobileNet V2	300	3.47

The picture above shows the numbers from MobileNet V1 and V2 belong to the model versions with 1.0 depth multiplier. It is better, if the numbers are lower in this table. By seeing the results we can assume that V2 is almost twice as fast as V1 model. On a mobile device when memory access is limited than the computational capability V2 works very well.

MACs—multiply-accumulate ...

Credit

Let Us Start!

```
# import libraries

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import os
import random
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import shutil
import tensorflow as tf
from keras.preprocessing.image import ImageDataGenerator
from keras.layers.core import Dense

# define the base working directory
```

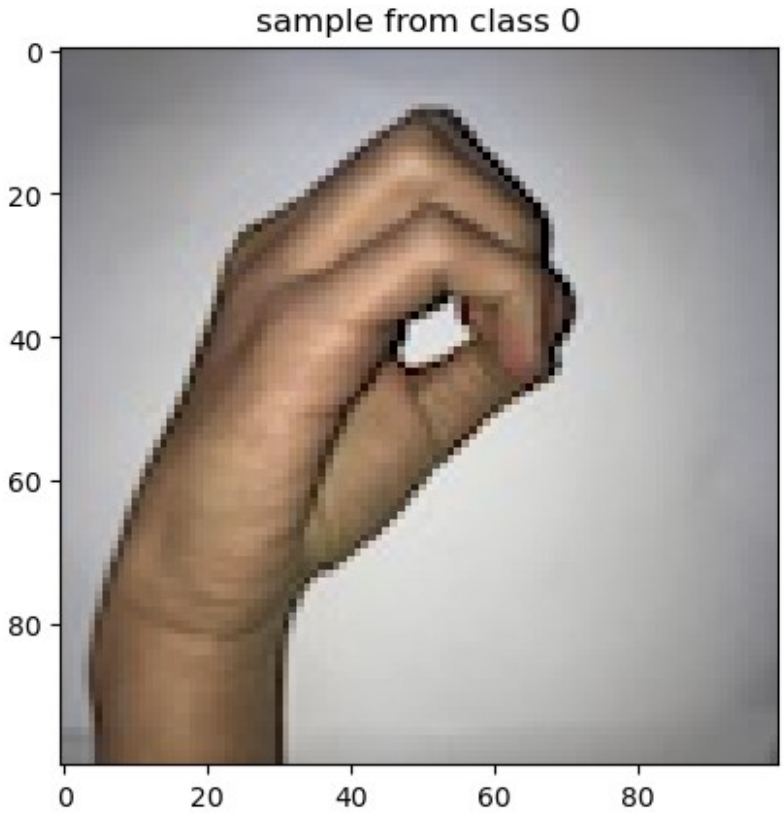
```
BASE_DIR = os.getcwd()

# define data directory

DATA_DIR = os.path.join(BASE_DIR, 'data')

# define paths to the other directories

TRAIN_DIR = os.path.join(DATA_DIR, 'train')
VALIDATION_DIR = os.path.join(DATA_DIR, 'validation')
TEST_DIR = os.path.join(DATA_DIR, 'test')
```

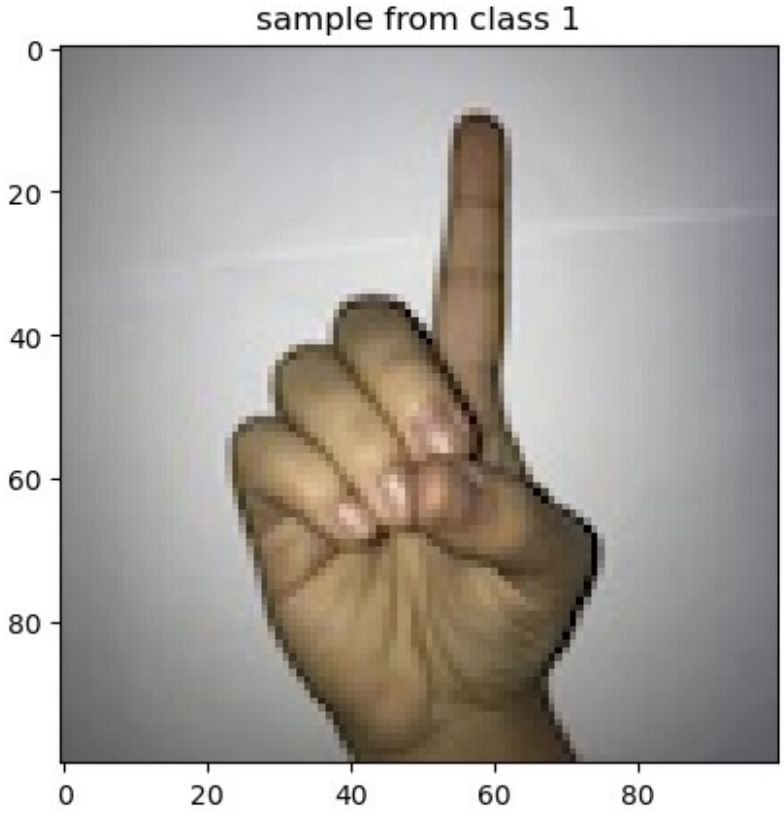
Details of datasets:

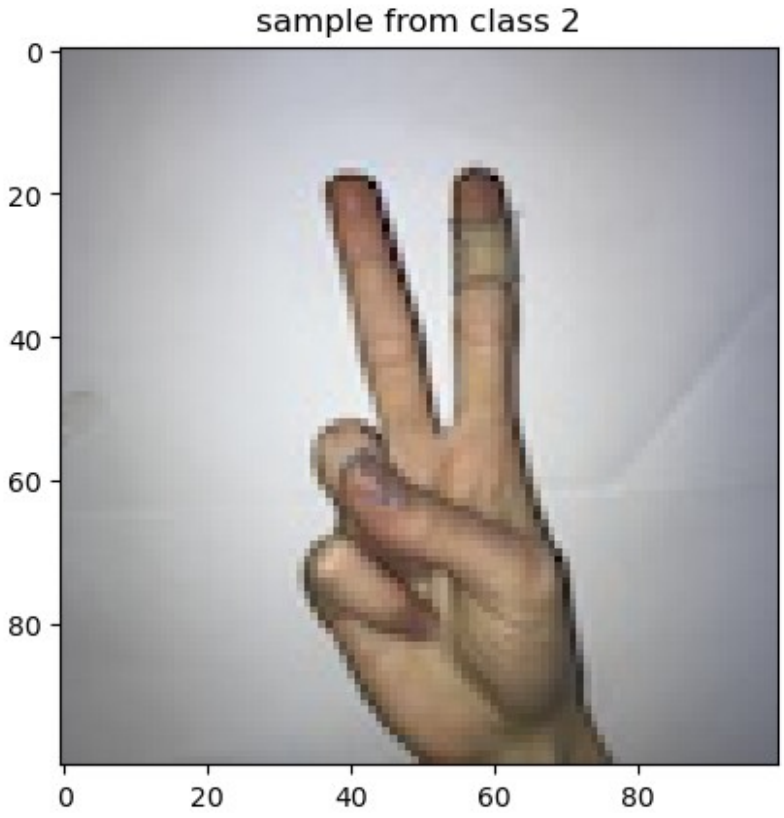
- Image size: 100 x 100 pixels
- Color space: RGB
- Number of classes: 10 (Digits: 0-9)
- Number of participant students: 218
- Number of samples per student: 10

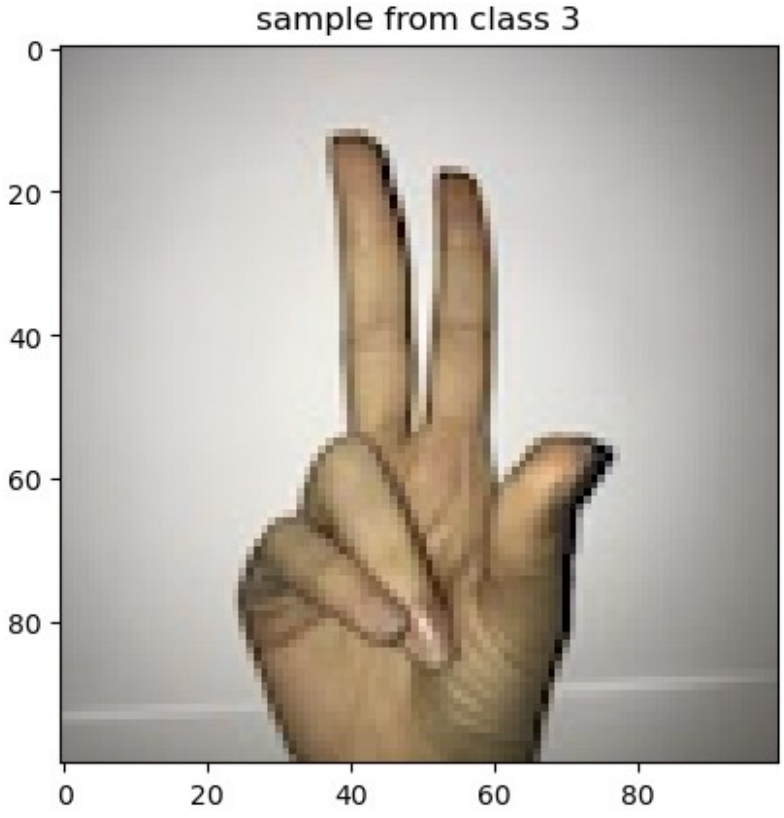
Data set Preview

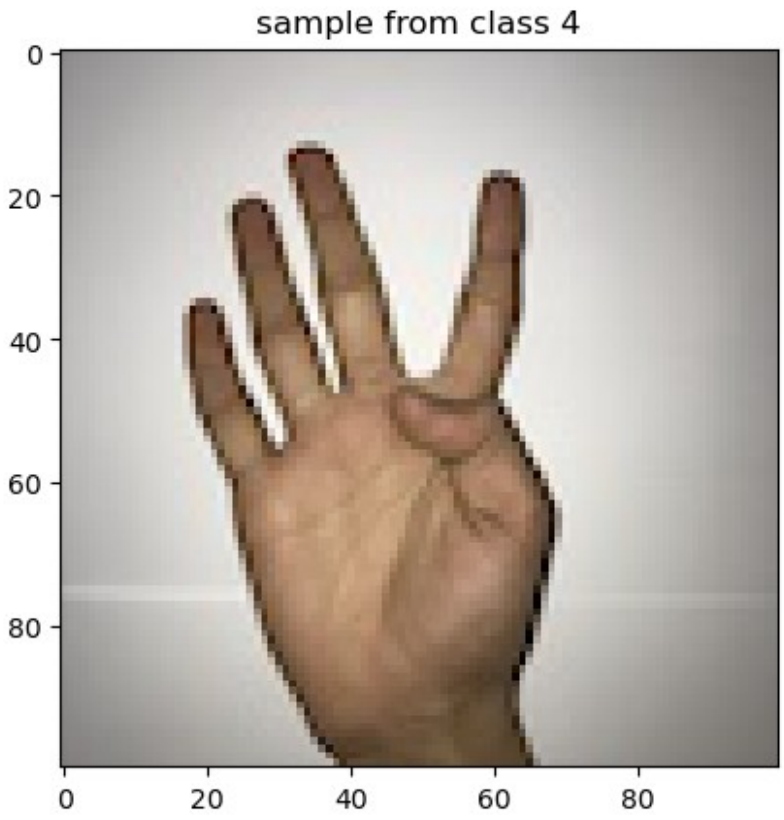
```
# A function to get a random image from each class using the data
folder, and show them


def show_nRandom_images(data_dir= DATA_DIR):
    # for each class, grab one random image
    for class_ in os.listdir(DATA_DIR):
        # Get a list of all image files in the folder
        image_files = [f for f in
os.listdir(os.path.join(DATA_DIR,class_)) if
os.path.isfile(os.path.join(DATA_DIR,class_ ,f))]
        # Select a random image from the list and Display it

        img =
mpimg.imread(os.path.join(os.path.join(DATA_DIR,class_ ,random.choice(
image_files))))
        plt.imshow(img)
        plt.title('sample from class {}'.format(class_))
#         plt.axis('off')
        plt.show()

# execute the function
show_nRandom_images()
```

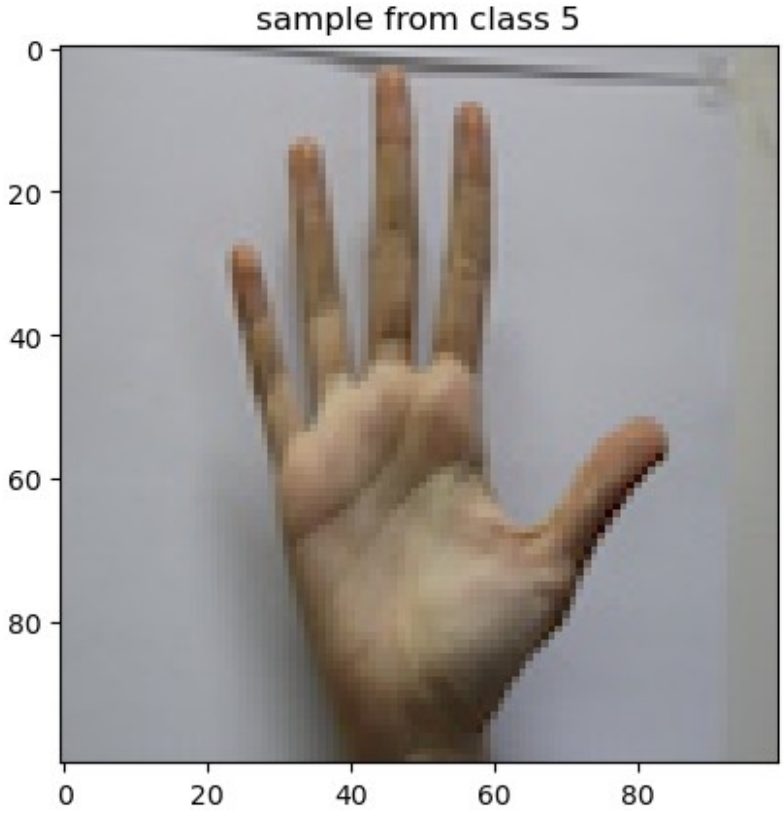

sample from class 0
0
20
40
60
80
0
20
40
60
80

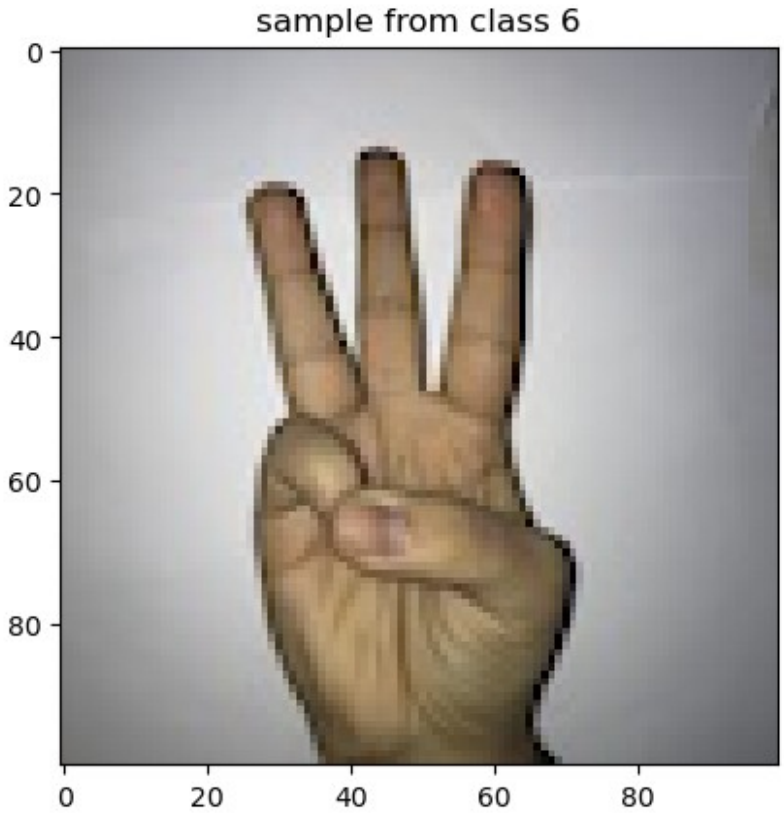

sample from class 1
0
20
40
60
80
0
20
40
60
80

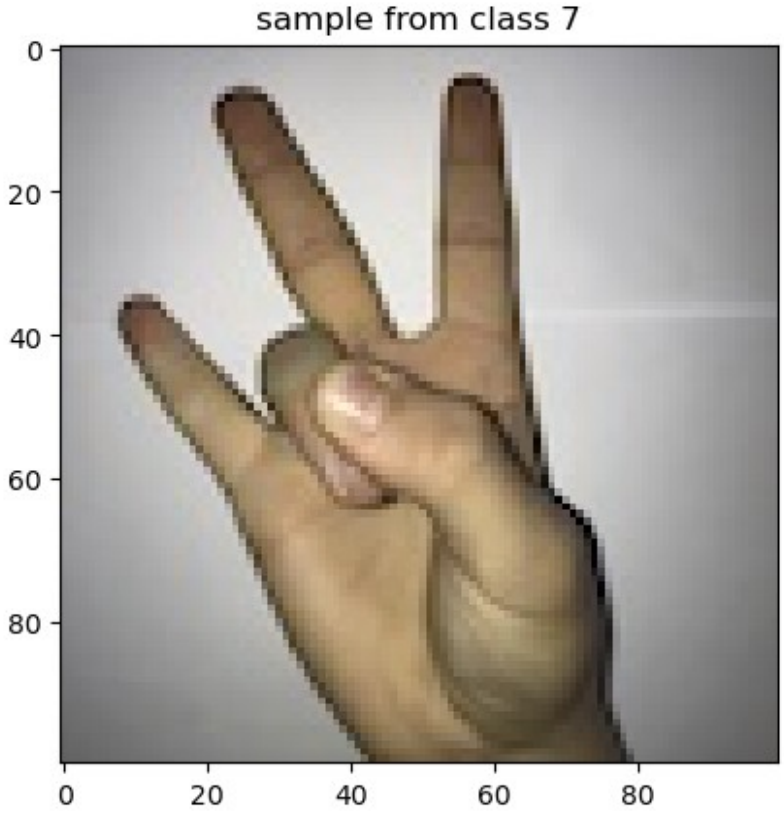

sample from class 2
0
20
40
60
80
0
20
40
60
80

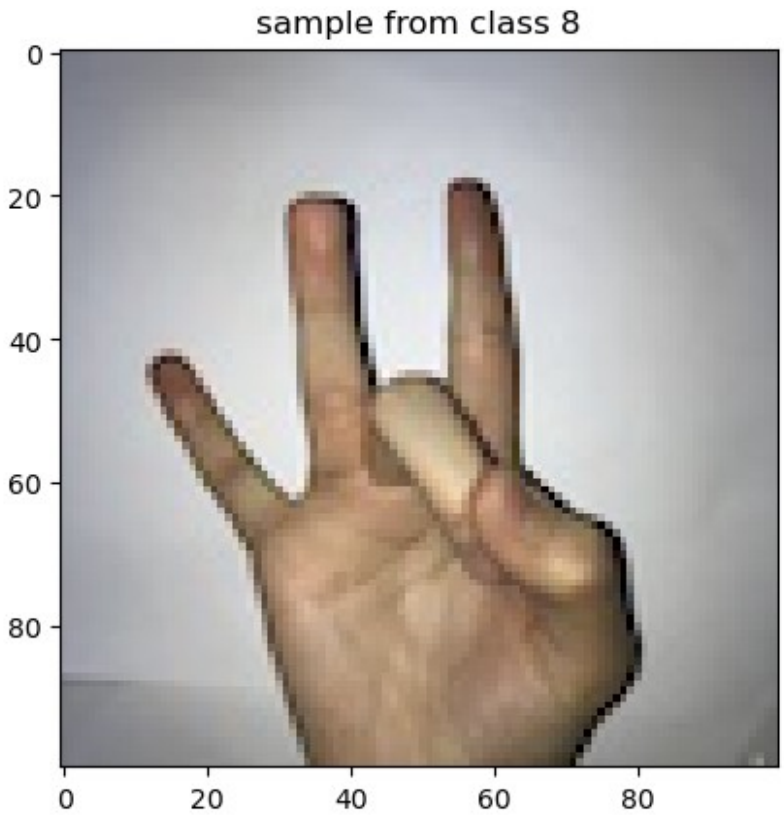

sample from class 3
0
20
40
60
80
0
20
40
60
80

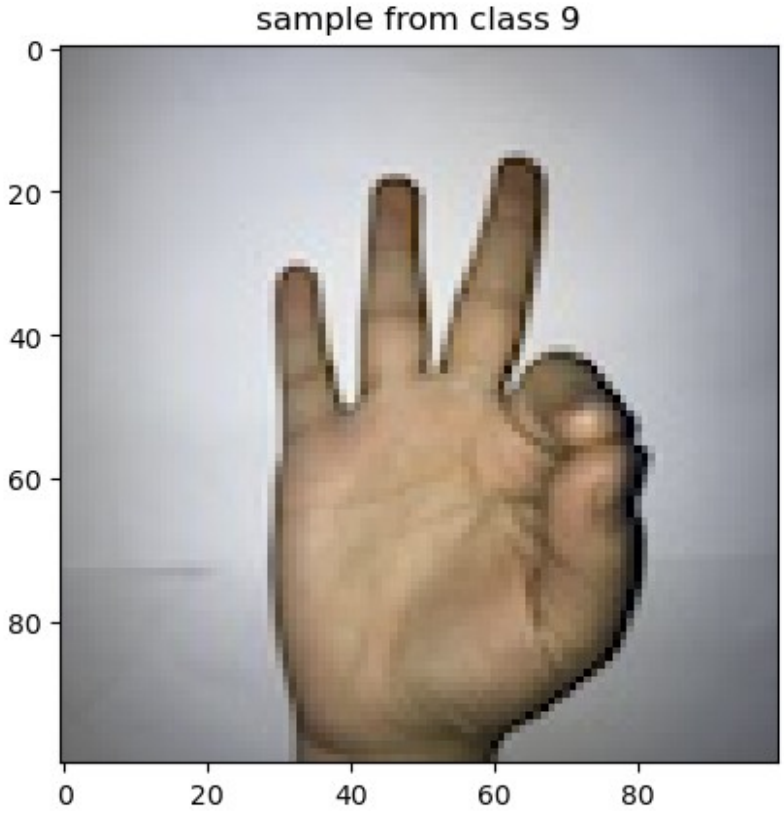

sample from class 4
0
20
40
60
80
0
20
40
60
80


sample from class 5
0
20
40
60
80
0
20
40
60
80


sample from class 6
0
20
40
60
80
0
20
40
60
80


sample from class 7
0
20
40
60
80
0
20
40
60
80


sample from class 8
0
20
40
60
80
0
20
40
60
80


sample from class 9
0
20
40
60
80
0
20
40
60
80

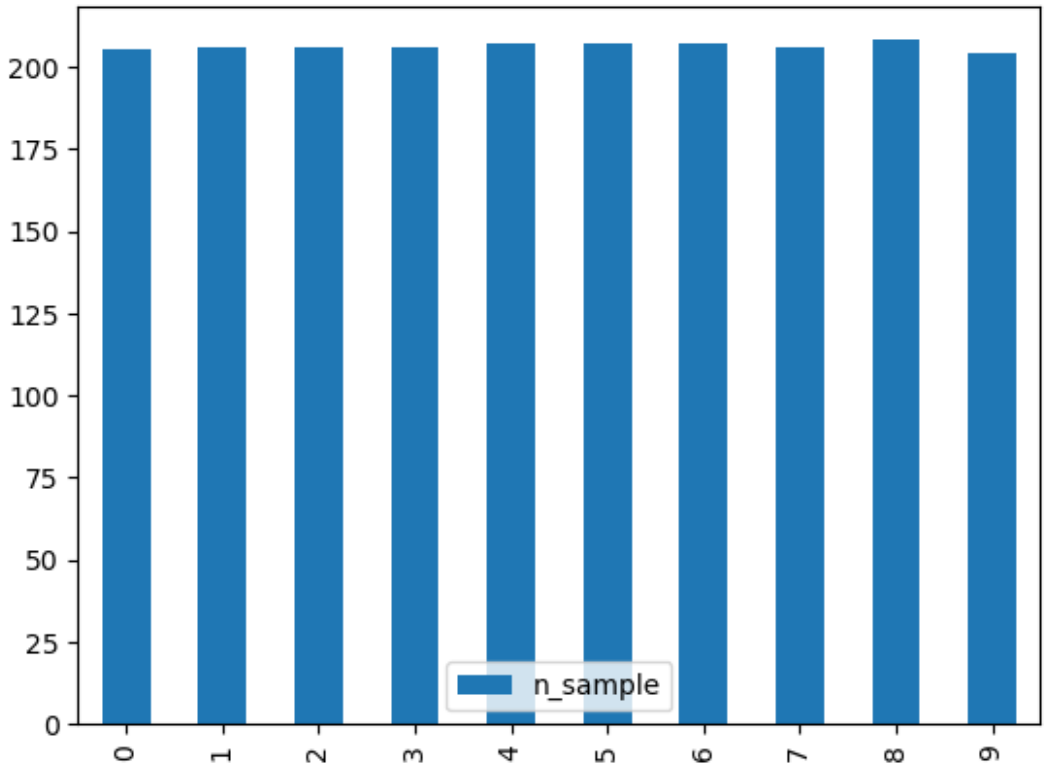
Statistics about the dataset

```
#### See number of samples within each class

def get_metadata():
    metadata_df = pd.DataFrame(columns=['class','n_sample'] ,
index=[])

    for class_ in range(10):
        # Get a list of all image files in the folder
        image_files = [f for f in
os.listdir(os.path.join(DATA_DIR,str(class_))) if
os.path.isfile(os.path.join(DATA_DIR,str(class_) ,f))]
        metadata_df.loc[metadata_df.shape[0],['class','n_sample']] =
[str(class_),len(image_files)]
    return metadata_df

# see results
get_metadata()
```

	class	n_sample
0	0	205
1	1	206
2	2	206
3	3	206
4	4	207
5	5	207
6	6	207
7	7	206
8	8	208
9	9	204

```
# more details about each class samples
get_metadata().plot.bar()
```

```
<AxesSubplot:>
```



The data is balanced as expected

Organize data into train, validation and test data

Our data folder will follow the given structure

data/ train/ (up to 170 samples) validation/ (30 samples) test/ (5 samples)

```
# verify that this structure is not here yet
if(os.path.isdir(os.path.join(DATA_DIR,'train')) == False):

    # create three directories (train, val, test)
    os.mkdir(os.path.join(DATA_DIR,'train'))
    os.mkdir(os.path.join(DATA_DIR,'validation'))
    os.mkdir(os.path.join(DATA_DIR,'test'))

    # Organize data to follow a machine learning dataset structure

        # loop through the original data folder
    for class_ in range(10):
        # Get a list of all image files in the folder
        image_files = [f for f in
os.listdir(os.path.join(DATA_DIR,str(class_))) if
os.path.isfile(os.path.join(DATA_DIR,str(class_) ,f))]

        # for validation
```

```
            # create corresponding class_ folder
        os.mkdir(os.path.join(VALIDATION_DIR,str(class_)))
            # take 30 random samples
        validation_set = random.sample(image_files, 30)
            # move them to the validation directory
        for validation_img in validation_set:

shutil.move(os.path.join(DATA_DIR,str(class_),validation_img),
os.path.join(VALIDATION_DIR,str(class_)))

        # for test

        # Refresh the list of all image files in the folder
        image_files = [f for f in
os.listdir(os.path.join(DATA_DIR,str(class_))) if
os.path.isfile(os.path.join(DATA_DIR,str(class_) ,f))]

            # create corresponding class_ folder
        os.mkdir(os.path.join(TEST_DIR,str(class_)))
            # take 30 random samples
        test_set = random.sample(image_files, 5)
            # move them to the test directory
        for test_img in test_set:
            shutil.move(os.path.join(DATA_DIR,str(class_),test_img),
os.path.join(TEST_DIR,str(class_)))

    # for training
        # takes the remaining samples

    for class_ in range(10):

        # Get a list of all image files in the folder
        train_set = [f for f in
os.listdir(os.path.join(DATA_DIR,str(class_))) if
os.path.isfile(os.path.join(DATA_DIR,str(class_) ,f))]

        # create corresponding class_ folder
        os.mkdir(os.path.join(TRAIN_DIR,str(class_)))

        # move them to the validation directory
        for train_img in train_set:
            shutil.move(os.path.join(DATA_DIR,str(class_),train_img),
os.path.join(TRAIN_DIR,str(class_)))


else:
    print('The structure you are trying to create is already here!')
```

Verify the new distribution of data

```
# a function to do the job

def get_metadataOfFolder(folder_name, folder_path):
    metadata_df =
pd.DataFrame(columns=[folder_name+'_classes','{}_samples'.format(folde
r_name)] , index=[])

    for class_ in os.listdir(folder_path):
        # Get a list of all image files in the folder
        image_files = [f for f in
os.listdir(os.path.join(folder_path,class_)) if
os.path.isfile(os.path.join(folder_path, class_ ,f))]
        metadata_df.loc[metadata_df.shape[0],
[folder_name+'_classes','{}_samples'.format(folder_name)]] =
[str(class_),len(image_files)]
    return metadata_df

# for training data
get_metadataOfFolder('train',TRAIN_DIR)
```

	train_classes	train_samples
0	0	170
1	1	171
2	2	171
3	3	171
4	4	172
5	5	172
6	6	172
7	7	171
8	8	173
9	9	169

```
# for testing data
get_metadataOfFolder('test',TEST_DIR)
```

	test_classes	test_samples
0	0	5
1	1	5
2	2	5
3	3	5
4	4	5
5	5	5
6	6	5
7	7	5
8	8	5
9	9	5

```
# for validation data
get_metadataOfFolder('validation',VALIDATION_DIR)
```

```
  validation_classes validation_samples
0                  0                 30
1                  1                 30
2                  2                 30
3                  3                 30
4                  4                 30
5                  5                 30
6                  6                 30
7                  7                 30
8                  8                 30
9                  9                 30
```

Pre-Process the data

- We will be converting each images to follow the same format MobileNet expects

```
train_chunks =
ImageDataGenerator(preprocessing_function=tf.keras.applications.mobile
net_v2.preprocess_input).flow_from_directory(
    directory=TRAIN_DIR, target_size=(224,224), batch_size=10)

valid_chunks =
ImageDataGenerator(preprocessing_function=tf.keras.applications.mobile
net_v2.preprocess_input).flow_from_directory(
    directory=VALIDATION_DIR, target_size=(224,224), batch_size=10)

test_chunks =
ImageDataGenerator(preprocessing_function=tf.keras.applications.mobile
net_v2.preprocess_input).flow_from_directory(
    directory=TEST_DIR, target_size=(224,224), batch_size=10,
shuffle=False)

Found 1712 images belonging to 10 classes.
Found 300 images belonging to 10 classes.
Found 50 images belonging to 10 classes.
```

Modeling part

```
# Initialize the MobileNetV2 object

mobilenetv2 = tf.keras.applications.MobileNetV2(weights='imagenet',
include_top=False, input_shape=(224, 224, 3))
```

By setting include_top=False, you exclude these dense layers, effectively removing the final classification/regression functionality of the pre-trained model. This allows you to add your own custom top layers suited to your specific task, such as a different number of output classes or a different type of prediction.

```
# Discover model architecture

mobilenetv2.summary()
```

```
Model: "mobilenetv2_1.00_224"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
Connected to
==================================================================
 input_17 (InputLayer)       [(None, 224, 224, 3  0           []

                             )]



 Conv1 (Conv2D)              (None, 112, 112, 32  864
['input_17[0][0]']
                             )



 bn_Conv1 (BatchNormalization)  (None, 112, 112, 32  128
['Conv1[0][0]']
                             )



 Conv1_relu (ReLU)           (None, 112, 112, 32  0
['bn_Conv1[0][0]']
                             )



 expanded_conv_depthwise (Depth  (None, 112, 112, 32  288
['Conv1_relu[0][0]']
 wiseConv2D)                 )



 expanded_conv_depthwise_BN (Ba  (None, 112, 112, 32  128
['expanded_conv_depthwise[0][0]']
 tchNormalization)           )



 expanded_conv_depthwise_relu (  (None, 112, 112, 32  0
['expanded_conv_depthwise_BN[0][0
 ReLU)                       )                                  ']
```

```
 expanded_conv_project (Conv2D)  (None, 112, 112, 16  512
['expanded_conv_depthwise_relu[0]
                                )                                 [0]']


 expanded_conv_project_BN (Batc  (None, 112, 112, 16  64
['expanded_conv_project[0][0]']
 hNormalization)                )


 block_1_expand (Conv2D)        (None, 112, 112, 96  1536
['expanded_conv_project_BN[0][0]'
                                )                                 ]


 block_1_expand_BN (BatchNormal  (None, 112, 112, 96  384
['block_1_expand[0][0]']
 ization)                       )


 block_1_expand_relu (ReLU)     (None, 112, 112, 96  0
['block_1_expand_BN[0][0]']
                                )


 block_1_pad (ZeroPadding2D)    (None, 113, 113, 96  0
['block_1_expand_relu[0][0]']
                                )


 block_1_depthwise (DepthwiseCo  (None, 56, 56, 96)  864
['block_1_pad[0][0]']
 nv2D)


 block_1_depthwise_BN (BatchNor  (None, 56, 56, 96)  384
['block_1_depthwise[0][0]']
 malization)


 block_1_depthwise_relu (ReLU)  (None, 56, 56, 96)   0
```

```
['block_1_depthwise_BN[0][0]']


 block_1_project (Conv2D)       (None, 56, 56, 24)   2304
['block_1_depthwise_relu[0][0]']


 block_1_project_BN (BatchNorma  (None, 56, 56, 24)  96
['block_1_project[0][0]']
 lization)



 block_2_expand (Conv2D)        (None, 56, 56, 144)  3456
['block_1_project_BN[0][0]']


 block_2_expand_BN (BatchNormal  (None, 56, 56, 144)  576
['block_2_expand[0][0]']
 ization)



 block_2_expand_relu (ReLU)     (None, 56, 56, 144)  0
['block_2_expand_BN[0][0]']


 block_2_depthwise (DepthwiseCo  (None, 56, 56, 144)  1296
['block_2_expand_relu[0][0]']
 nv2D)



 block_2_depthwise_BN (BatchNor  (None, 56, 56, 144)  576
['block_2_depthwise[0][0]']
 malization)



 block_2_depthwise_relu (ReLU)  (None, 56, 56, 144)  0
['block_2_depthwise_BN[0][0]']


 block_2_project (Conv2D)       (None, 56, 56, 24)   3456
['block_2_depthwise_relu[0][0]']


 block_2_project_BN (BatchNorma  (None, 56, 56, 24)  96
['block_2_project[0][0]']
 lization)
```

```
 block_2_add (Add)               (None, 56, 56, 24)   0
['block_1_project_BN[0][0]',

'block_2_project_BN[0][0]']


 block_3_expand (Conv2D)         (None, 56, 56, 144)  3456
['block_2_add[0][0]']


 block_3_expand_BN (BatchNormal  (None, 56, 56, 144)  576
['block_3_expand[0][0]']
 ization)



 block_3_expand_relu (ReLU)      (None, 56, 56, 144)  0
['block_3_expand_BN[0][0]']


 block_3_pad (ZeroPadding2D)     (None, 57, 57, 144)  0
['block_3_expand_relu[0][0]']


 block_3_depthwise (DepthwiseCo  (None, 28, 28, 144)  1296
['block_3_pad[0][0]']
 nv2D)



 block_3_depthwise_BN (BatchNor  (None, 28, 28, 144)  576
['block_3_depthwise[0][0]']
 malization)



 block_3_depthwise_relu (ReLU)  (None, 28, 28, 144)  0
['block_3_depthwise_BN[0][0]']


 block_3_project (Conv2D)        (None, 28, 28, 32)   4608
['block_3_depthwise_relu[0][0]']


 block_3_project_BN (BatchNorma  (None, 28, 28, 32)  128
['block_3_project[0][0]']
 lization)
```

```
 block_4_expand (Conv2D)        (None, 28, 28, 192)  6144
['block_3_project_BN[0][0]']


 block_4_expand_BN (BatchNormal  (None, 28, 28, 192)  768
['block_4_expand[0][0]']
 ization)



 block_4_expand_relu (ReLU)     (None, 28, 28, 192)  0
['block_4_expand_BN[0][0]']


 block_4_depthwise (DepthwiseCo  (None, 28, 28, 192)  1728
['block_4_expand_relu[0][0]']
 nv2D)



 block_4_depthwise_BN (BatchNor  (None, 28, 28, 192)  768
['block_4_depthwise[0][0]']
 malization)



 block_4_depthwise_relu (ReLU)  (None, 28, 28, 192)  0
['block_4_depthwise_BN[0][0]']


 block_4_project (Conv2D)       (None, 28, 28, 32)   6144
['block_4_depthwise_relu[0][0]']


 block_4_project_BN (BatchNorma  (None, 28, 28, 32)  128
['block_4_project[0][0]']
 lization)



 block_4_add (Add)              (None, 28, 28, 32)   0
['block_3_project_BN[0][0]',

'block_4_project_BN[0][0]']


 block_5_expand (Conv2D)        (None, 28, 28, 192)  6144
['block_4_add[0][0]']
```

```
 block_5_expand_BN (BatchNormal  (None, 28, 28, 192)  768
['block_5_expand[0][0]']
 ization)



 block_5_expand_relu (ReLU)     (None, 28, 28, 192)  0
['block_5_expand_BN[0][0]']


 block_5_depthwise (DepthwiseCo  (None, 28, 28, 192)  1728
['block_5_expand_relu[0][0]']
 nv2D)



 block_5_depthwise_BN (BatchNor  (None, 28, 28, 192)  768
['block_5_depthwise[0][0]']
 malization)



 block_5_depthwise_relu (ReLU)  (None, 28, 28, 192)  0
['block_5_depthwise_BN[0][0]']


 block_5_project (Conv2D)       (None, 28, 28, 32)   6144
['block_5_depthwise_relu[0][0]']


 block_5_project_BN (BatchNorma  (None, 28, 28, 32)  128
['block_5_project[0][0]']
 lization)



 block_5_add (Add)              (None, 28, 28, 32)   0
['block_4_add[0][0]',

'block_5_project_BN[0][0]']


 block_6_expand (Conv2D)        (None, 28, 28, 192)  6144
['block_5_add[0][0]']


 block_6_expand_BN (BatchNormal  (None, 28, 28, 192)  768
['block_6_expand[0][0]']
 ization)
```

```
 block_6_expand_relu (ReLU)     (None, 28, 28, 192)  0
['block_6_expand_BN[0][0]']


 block_6_pad (ZeroPadding2D)    (None, 29, 29, 192)  0
['block_6_expand_relu[0][0]']


 block_6_depthwise (DepthwiseCo  (None, 14, 14, 192)  1728
['block_6_pad[0][0]']
 nv2D)



 block_6_depthwise_BN (BatchNor  (None, 14, 14, 192)  768
['block_6_depthwise[0][0]']
 malization)



 block_6_depthwise_relu (ReLU)  (None, 14, 14, 192)  0
['block_6_depthwise_BN[0][0]']


 block_6_project (Conv2D)       (None, 14, 14, 64)   12288
['block_6_depthwise_relu[0][0]']


 block_6_project_BN (BatchNorma  (None, 14, 14, 64)  256
['block_6_project[0][0]']

 lization)



 block_7_expand (Conv2D)        (None, 14, 14, 384)  24576
['block_6_project_BN[0][0]']


 block_7_expand_BN (BatchNormal  (None, 14, 14, 384)  1536
['block_7_expand[0][0]']
 ization)



 block_7_expand_relu (ReLU)     (None, 14, 14, 384)  0
['block_7_expand_BN[0][0]']


 block_7_depthwise (DepthwiseCo  (None, 14, 14, 384)  3456
```

```
['block_7_expand_relu[0][0]']
 nv2D)


 block_7_depthwise_BN (BatchNor  (None, 14, 14, 384)  1536
['block_7_depthwise[0][0]']
 malization)


 block_7_depthwise_relu (ReLU)  (None, 14, 14, 384)  0
['block_7_depthwise_BN[0][0]']


 block_7_project (Conv2D)       (None, 14, 14, 64)   24576
['block_7_depthwise_relu[0][0]']


 block_7_project_BN (BatchNorma  (None, 14, 14, 64)  256
['block_7_project[0][0]']
 lization)


 block_7_add (Add)              (None, 14, 14, 64)   0
['block_6_project_BN[0][0]',

'block_7_project_BN[0][0]']


 block_8_expand (Conv2D)        (None, 14, 14, 384)  24576
['block_7_add[0][0]']


 block_8_expand_BN (BatchNormal  (None, 14, 14, 384)  1536
['block_8_expand[0][0]']
 ization)


 block_8_expand_relu (ReLU)     (None, 14, 14, 384)  0
['block_8_expand_BN[0][0]']


 block_8_depthwise (DepthwiseCo  (None, 14, 14, 384)  3456
['block_8_expand_relu[0][0]']
 nv2D)
```

```
 block_8_depthwise_BN (BatchNor  (None, 14, 14, 384)  1536
['block_8_depthwise[0][0]']
 malization)


 block_8_depthwise_relu (ReLU)  (None, 14, 14, 384)  0
['block_8_depthwise_BN[0][0]']


 block_8_project (Conv2D)       (None, 14, 14, 64)   24576
['block_8_depthwise_relu[0][0]']


 block_8_project_BN (BatchNorma  (None, 14, 14, 64)  256
['block_8_project[0][0]']
 lization)


 block_8_add (Add)              (None, 14, 14, 64)   0
['block_7_add[0][0]',

'block_8_project_BN[0][0]']


 block_9_expand (Conv2D)        (None, 14, 14, 384)  24576
['block_8_add[0][0]']


 block_9_expand_BN (BatchNormal  (None, 14, 14, 384)  1536
['block_9_expand[0][0]']
 ization)


 block_9_expand_relu (ReLU)     (None, 14, 14, 384)  0
['block_9_expand_BN[0][0]']


 block_9_depthwise (DepthwiseCo  (None, 14, 14, 384)  3456
['block_9_expand_relu[0][0]']
 nv2D)


 block_9_depthwise_BN (BatchNor  (None, 14, 14, 384)  1536
['block_9_depthwise[0][0]']
 malization)
```

```
 block_9_depthwise_relu (ReLU)  (None, 14, 14, 384)  0
['block_9_depthwise_BN[0][0]']


 block_9_project (Conv2D)       (None, 14, 14, 64)   24576
['block_9_depthwise_relu[0][0]']


 block_9_project_BN (BatchNorma  (None, 14, 14, 64)  256
['block_9_project[0][0]']
 lization)



 block_9_add (Add)              (None, 14, 14, 64)   0
['block_8_add[0][0]',

'block_9_project_BN[0][0]']


 block_10_expand (Conv2D)       (None, 14, 14, 384)  24576
['block_9_add[0][0]']


 block_10_expand_BN (BatchNorma  (None, 14, 14, 384)  1536
['block_10_expand[0][0]']
 lization)



 block_10_expand_relu (ReLU)    (None, 14, 14, 384)  0
['block_10_expand_BN[0][0]']


 block_10_depthwise (DepthwiseC  (None, 14, 14, 384)  3456
['block_10_expand_relu[0][0]']
 onv2D)



 block_10_depthwise_BN (BatchNo  (None, 14, 14, 384)  1536
['block_10_depthwise[0][0]']
 rmalization)



 block_10_depthwise_relu (ReLU)  (None, 14, 14, 384)  0
['block_10_depthwise_BN[0][0]']
```

```
 block_10_project (Conv2D)      (None, 14, 14, 96)   36864
['block_10_depthwise_relu[0][0]']


 block_10_project_BN (BatchNorm  (None, 14, 14, 96)  384
['block_10_project[0][0]']
 alization)



 block_11_expand (Conv2D)       (None, 14, 14, 576)  55296
['block_10_project_BN[0][0]']


 block_11_expand_BN (BatchNorma  (None, 14, 14, 576)  2304
['block_11_expand[0][0]']
 lization)



 block_11_expand_relu (ReLU)    (None, 14, 14, 576)  0
['block_11_expand_BN[0][0]']


 block_11_depthwise (DepthwiseC  (None, 14, 14, 576)  5184
['block_11_expand_relu[0][0]']
 onv2D)



 block_11_depthwise_BN (BatchNo  (None, 14, 14, 576)  2304
['block_11_depthwise[0][0]']
 rmalization)



 block_11_depthwise_relu (ReLU)  (None, 14, 14, 576)  0
['block_11_depthwise_BN[0][0]']


 block_11_project (Conv2D)      (None, 14, 14, 96)   55296
['block_11_depthwise_relu[0][0]']


 block_11_project_BN (BatchNorm  (None, 14, 14, 96)  384
['block_11_project[0][0]']
 alization)



 block_11_add (Add)             (None, 14, 14, 96)   0
```

```
['block_10_project_BN[0][0]',

'block_11_project_BN[0][0]']


 block_12_expand (Conv2D)       (None, 14, 14, 576)  55296
['block_11_add[0][0]']


 block_12_expand_BN (BatchNorma  (None, 14, 14, 576)  2304
['block_12_expand[0][0]']
 lization)



 block_12_expand_relu (ReLU)    (None, 14, 14, 576)  0
['block_12_expand_BN[0][0]']


 block_12_depthwise (DepthwiseC  (None, 14, 14, 576)  5184
['block_12_expand_relu[0][0]']
 onv2D)



 block_12_depthwise_BN (BatchNo  (None, 14, 14, 576)  2304
['block_12_depthwise[0][0]']
 rmalization)



 block_12_depthwise_relu (ReLU)  (None, 14, 14, 576)  0
['block_12_depthwise_BN[0][0]']


 block_12_project (Conv2D)      (None, 14, 14, 96)   55296
['block_12_depthwise_relu[0][0]']


 block_12_project_BN (BatchNorm  (None, 14, 14, 96)  384
['block_12_project[0][0]']
 alization)



 block_12_add (Add)             (None, 14, 14, 96)   0
['block_11_add[0][0]',

'block_12_project_BN[0][0]']
```

```
 block_13_expand (Conv2D)       (None, 14, 14, 576)  55296
['block_12_add[0][0]']


 block_13_expand_BN (BatchNorma  (None, 14, 14, 576)  2304
['block_13_expand[0][0]']
 lization)



 block_13_expand_relu (ReLU)    (None, 14, 14, 576)  0
['block_13_expand_BN[0][0]']


 block_13_pad (ZeroPadding2D)   (None, 15, 15, 576)  0
['block_13_expand_relu[0][0]']


 block_13_depthwise (DepthwiseC  (None, 7, 7, 576)   5184
['block_13_pad[0][0]']
 onv2D)



 block_13_depthwise_BN (BatchNo  (None, 7, 7, 576)   2304
['block_13_depthwise[0][0]']
 rmalization)



 block_13_depthwise_relu (ReLU)  (None, 7, 7, 576)   0
['block_13_depthwise_BN[0][0]']


 block_13_project (Conv2D)      (None, 7, 7, 160)    92160
['block_13_depthwise_relu[0][0]']


 block_13_project_BN (BatchNorm  (None, 7, 7, 160)   640
['block_13_project[0][0]']
 alization)



 block_14_expand (Conv2D)       (None, 7, 7, 960)    153600
['block_13_project_BN[0][0]']


 block_14_expand_BN (BatchNorma  (None, 7, 7, 960)   3840
['block_14_expand[0][0]']
 lization)
```

```
 block_14_expand_relu (ReLU)    (None, 7, 7, 960)    0
['block_14_expand_BN[0][0]']


 block_14_depthwise (DepthwiseC  (None, 7, 7, 960)   8640
['block_14_expand_relu[0][0]']
 onv2D)



 block_14_depthwise_BN (BatchNo  (None, 7, 7, 960)   3840
['block_14_depthwise[0][0]']
 rmalization)



 block_14_depthwise_relu (ReLU)  (None, 7, 7, 960)   0
['block_14_depthwise_BN[0][0]']


 block_14_project (Conv2D)      (None, 7, 7, 160)    153600
['block_14_depthwise_relu[0][0]']


 block_14_project_BN (BatchNorm  (None, 7, 7, 160)   640
['block_14_project[0][0]']
 alization)



 block_14_add (Add)             (None, 7, 7, 160)    0
['block_13_project_BN[0][0]',

'block_14_project_BN[0][0]']


 block_15_expand (Conv2D)       (None, 7, 7, 960)    153600
['block_14_add[0][0]']


 block_15_expand_BN (BatchNorma  (None, 7, 7, 960)   3840
['block_15_expand[0][0]']
 lization)



 block_15_expand_relu (ReLU)    (None, 7, 7, 960)    0
['block_15_expand_BN[0][0]']
```

```
 block_15_depthwise (DepthwiseC  (None, 7, 7, 960)   8640
['block_15_expand_relu[0][0]']
 onv2D)


 block_15_depthwise_BN (BatchNo  (None, 7, 7, 960)   3840
['block_15_depthwise[0][0]']
 rmalization)


 block_15_depthwise_relu (ReLU)  (None, 7, 7, 960)   0
['block_15_depthwise_BN[0][0]']


 block_15_project (Conv2D)      (None, 7, 7, 160)    153600
['block_15_depthwise_relu[0][0]']


 block_15_project_BN (BatchNorm  (None, 7, 7, 160)   640
['block_15_project[0][0]']
 alization)


 block_15_add (Add)             (None, 7, 7, 160)    0
['block_14_add[0][0]',

'block_15_project_BN[0][0]']


 block_16_expand (Conv2D)       (None, 7, 7, 960)    153600
['block_15_add[0][0]']


 block_16_expand_BN (BatchNorma  (None, 7, 7, 960)   3840
['block_16_expand[0][0]']
 lization)


 block_16_expand_relu (ReLU)    (None, 7, 7, 960)    0
['block_16_expand_BN[0][0]']


 block_16_depthwise (DepthwiseC  (None, 7, 7, 960)   8640
['block_16_expand_relu[0][0]']
 onv2D)
```

```
 block_16_depthwise_BN (BatchNo  (None, 7, 7, 960)   3840
['block_16_depthwise[0][0]']
 rmalization)


 block_16_depthwise_relu (ReLU)  (None, 7, 7, 960)   0
['block_16_depthwise_BN[0][0]']


 block_16_project (Conv2D)      (None, 7, 7, 320)    307200
['block_16_depthwise_relu[0][0]']


 block_16_project_BN (BatchNorm  (None, 7, 7, 320)   1280
['block_16_project[0][0]']
 alization)


 Conv_1 (Conv2D)                (None, 7, 7, 1280)   409600
['block_16_project_BN[0][0]']


 Conv_1_bn (BatchNormalization)  (None, 7, 7, 1280)  5120
['Conv_1[0][0]']


 out_relu (ReLU)                (None, 7, 7, 1280)   0
['Conv_1_bn[0][0]']


==================================================================================
============================
Total params: 2,257,984
Trainable params: 2,223,872
Non-trainable params: 34,112
__________________________________________________________________________________
_____________________________
```

```
#Freeze the base model layers
mobilenetv2.trainable = False

# Create the input tensor
inputs = tf.keras.Input(shape=(224, 224, 3))

# Apply the pre-trained base model to the inputs
x = mobilenetv2(inputs, training=False)
```

```
# Add a global average pooling layer
x = tf.keras.layers.GlobalAveragePooling2D()(x)

# Add the output layer with the desired number of classes
outputs = Dense(units=10, activation='softmax')(x)

# Create the final model
model_ = tf.keras.Model(inputs, outputs)

# get the new model summary

model_.summary()
```

```
Model: "model_7"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_16 (InputLayer)       [(None, 224, 224, 3)]     0

 mobilenetv2_1.00_224 (Funct  (None, 7, 7, 1280)       2257984
 ional)

 global_average_pooling2d_11  (None, 1280)             0
  (GlobalAveragePooling2D)

 dense_15 (Dense)            (None, 10)                12810

=================================================================
Total params: 2,270,794
Trainable params: 12,810
Non-trainable params: 2,257,984
_________________________________________________________________
```

Train the model

```
# using a learning rate of 0.0001 and Adam optimizer

model_.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.0001
), loss='categorical_crossentropy', metrics=['accuracy'])

# Running the model using 10 epochs and saving the history

history = model_.fit(x=train_chunks,
            steps_per_epoch=len(train_chunks), # determines how many
batches are processed in one epoch,
            validation_data=valid_chunks,
            validation_steps=len(valid_chunks),
            epochs=30,
```

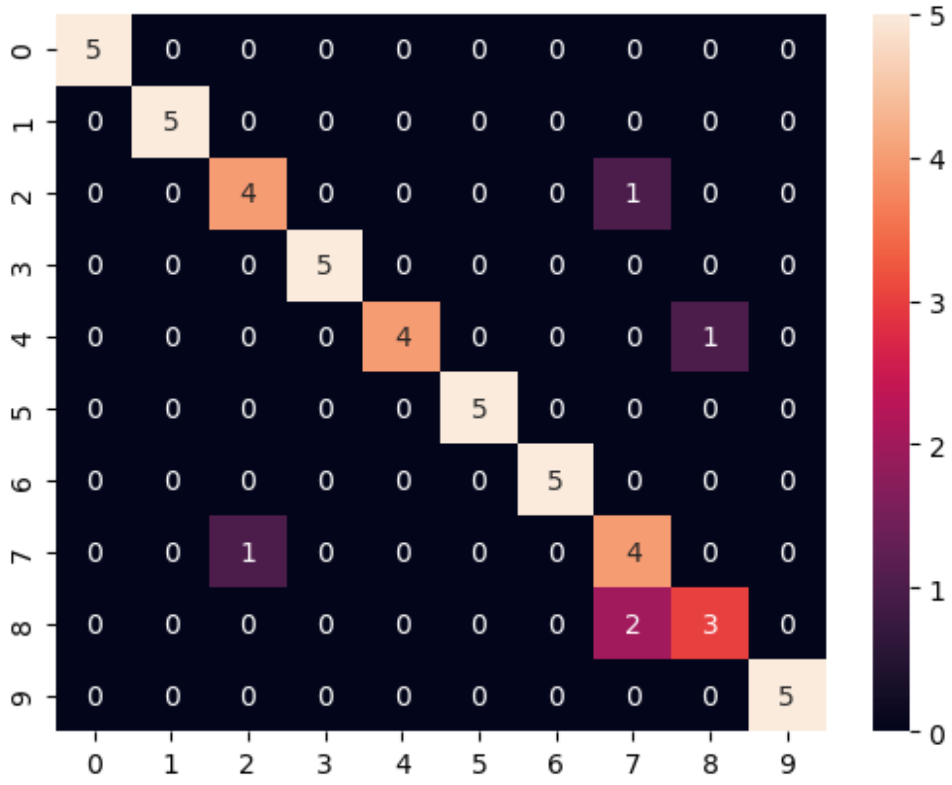
```
            verbose=2
)

Epoch 1/20
172/172 - 78s - loss: 0.4630 - accuracy: 0.8984 - val_loss: 0.4560 -
val_accuracy: 0.9067 - 78s/epoch - 455ms/step
Epoch 2/20
172/172 - 78s - loss: 0.4327 - accuracy: 0.9042 - val_loss: 0.4278 -
val_accuracy: 0.9067 - 78s/epoch - 454ms/step
Epoch 3/20
172/172 - 78s - loss: 0.4065 - accuracy: 0.9136 - val_loss: 0.4069 -
val_accuracy: 0.9167 - 78s/epoch - 455ms/step
Epoch 4/20
172/172 - 78s - loss: 0.3862 - accuracy: 0.9206 - val_loss: 0.3906 -
val_accuracy: 0.9100 - 78s/epoch - 456ms/step
Epoch 5/20
172/172 - 79s - loss: 0.3658 - accuracy: 0.9188 - val_loss: 0.3765 -
val_accuracy: 0.9133 - 79s/epoch - 458ms/step
Epoch 6/20
172/172 - 79s - loss: 0.3479 - accuracy: 0.9299 - val_loss: 0.3660 -
val_accuracy: 0.9233 - 79s/epoch - 458ms/step
Epoch 7/20
172/172 - 78s - loss: 0.3312 - accuracy: 0.9317 - val_loss: 0.3473 -
val_accuracy: 0.9300 - 78s/epoch - 453ms/step
Epoch 8/20
172/172 - 322s - loss: 0.3153 - accuracy: 0.9369 - val_loss: 0.3459 -
val_accuracy: 0.9233 - 322s/epoch - 2s/step
Epoch 9/20
172/172 - 82s - loss: 0.3036 - accuracy: 0.9363 - val_loss: 0.3288 -
val_accuracy: 0.9267 - 82s/epoch - 478ms/step
Epoch 10/20
172/172 - 79s - loss: 0.2901 - accuracy: 0.9422 - val_loss: 0.3143 -
val_accuracy: 0.9333 - 79s/epoch - 461ms/step
Epoch 11/20
172/172 - 78s - loss: 0.2784 - accuracy: 0.9457 - val_loss: 0.3015 -
val_accuracy: 0.9400 - 78s/epoch - 454ms/step
Epoch 12/20
172/172 - 78s - loss: 0.2673 - accuracy: 0.9509 - val_loss: 0.3003 -
val_accuracy: 0.9367 - 78s/epoch - 454ms/step
Epoch 13/20
172/172 - 79s - loss: 0.2583 - accuracy: 0.9486 - val_loss: 0.2924 -
val_accuracy: 0.9367 - 79s/epoch - 459ms/step
Epoch 14/20
172/172 - 79s - loss: 0.2477 - accuracy: 0.9544 - val_loss: 0.2830 -
val_accuracy: 0.9333 - 79s/epoch - 457ms/step
Epoch 15/20
172/172 - 79s - loss: 0.2394 - accuracy: 0.9539 - val_loss: 0.2807 -
val_accuracy: 0.9333 - 79s/epoch - 459ms/step
Epoch 16/20
172/172 - 79s - loss: 0.2311 - accuracy: 0.9579 - val_loss: 0.2697 -
```

```
val_accuracy: 0.9433 - 79s/epoch - 457ms/step
Epoch 17/20
172/172 - 79s - loss: 0.2237 - accuracy: 0.9550 - val_loss: 0.2663 -
val_accuracy: 0.9433 - 79s/epoch - 458ms/step
Epoch 18/20
172/172 - 78s - loss: 0.2162 - accuracy: 0.9603 - val_loss: 0.2645 -
val_accuracy: 0.9333 - 78s/epoch - 451ms/step
Epoch 19/20
172/172 - 79s - loss: 0.2087 - accuracy: 0.9620 - val_loss: 0.2624 -
val_accuracy: 0.9333 - 79s/epoch - 459ms/step
Epoch 20/20
172/172 - 79s - loss: 0.2025 - accuracy: 0.9638 - val_loss: 0.2556 -
val_accuracy: 0.9433 - 79s/epoch - 457ms/step
```

Inference

```
# real classes
test_labels = test_chunks.classes

# inference
predictions = model_.predict(x=test_chunks, steps=len(test_chunks),
verbose=0)

# confusion matrix for more details
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_true=test_labels,
y_pred=predictions.argmax(axis=1))

# plot the confusion matrix
import seaborn as sns
sns.heatmap(cm, annot=True)
```

```
<AxesSubplot:>
```




```
### Save the model to your local disk

model_.save('SignDigitsModel.h5')
```

- By: Ismail Ouahbi
- 06/07/2023